AED2 - Algoritmos e Estr. de Dados II

Aula 23: Coloração de Grafos

Prof. Aléssio Miranda Júnior
alessio@cefetmg.br

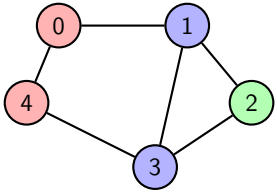
CEFET-MG - Campus Timoteo
Dep. Engenharia de Computação

Fevereiro de 2026

Sumário

1. O Mapa Mundi e a Coloração

Pegue um mapa político. A regra é simples: Países vizinhos devem ter cores diferentes. Qual o número **mínimo** de cores necessárias para pintar todo o mapa?



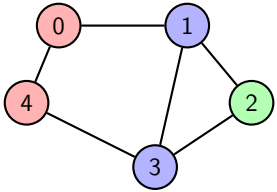
Arestas forçam cores distintas.
$\chi(G) = 3$ (3 cores)

1. O Mapa Mundi e a Coloração

Pegue um mapa político. A regra é simples: Países vizinhos devem ter cores diferentes. Qual o número **mínimo** de cores necessárias para pintar todo o mapa?

Definição Formal: k-coloração de Vértices

Uma atribuição de cores $c : V \to \{1 \dots k\}$ tal que $c(u) \neq c(v)$ para toda aresta $(u, v) \in E$.



Arestas forçam cores distintas.
$\chi(G) = 3$ (3 cores)

1. O Mapa Mundi e a Coloração

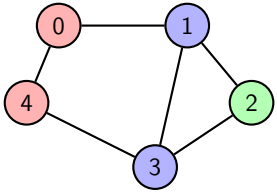
Pegue um mapa político. A regra é simples: Países vizinhos devem ter cores diferentes. Qual o número **mínimo** de cores necessárias para pintar todo o mapa?

Definição Formal: k-coloração de Vértices

Uma atribuição de cores $c : V \to \{1 \dots k\}$ tal que $c(u) \neq c(v)$ para toda aresta $(u, v) \in E$.

Número Cromático $\chi(G)$

É o **menor** valor de k para o qual existe uma k-coloração válida.



Arestas forçam cores distintas.
$\chi(G) = 3$ (3 cores)

2. Casos Básicos e Notáveis

Conhecer o número cromático de grafos notáveis nos ajuda a entender a complexidade do problema:

2. Casos Básicos e Notáveis

Conhecer o número cromático de grafos notáveis nos ajuda a entender a complexidade do problema:

- **Grafo Vazio (sem arestas):** $\chi(G) = 1$.
  Todos os vértices recebem a mesma cor.

2. Casos Básicos e Notáveis

Conhecer o número cromático de grafos notáveis nos ajuda a entender a complexidade do problema:

- **Grafo Vazio (sem arestas):** $\chi(G) = 1$. Todos os vértices recebem a mesma cor.
- **Grafo Bipartido:** $\chi(G) = 2$.

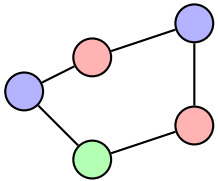
2. Casos Básicos e Notáveis

Conhecer o número cromático de grafos notáveis nos ajuda a entender a complexidade do problema:

- **Grafo Vazio (sem arestas):** $\chi(G) = 1$. Todos os vértices recebem a mesma cor.
- **Grafo Bipartido:** $\chi(G) = 2$.
- **Grafo Completo K_n:** $\chi(K_n) = n$. Como todos estão conectados entre si, nenhum vértice pode compartilhar a cor com outro.

Conhecer o número cromático de grafos notáveis nos ajuda a entender a complexidade do problema:

- **Grafo Vazio (sem arestas):** $\chi(G) = 1$. Todos os vértices recebem a mesma cor.
- **Grafo Bipartido:** $\chi(G) = 2$.
- **Grafo Completo K_n:** $\chi(K_n) = n$. Como todos estão conectados entre si, nenhum vértice pode compartilhar a cor com outro.
- **Ciclos:**
  - Ciclo Par C_{2k}: $\chi = 2$.
  - Ciclo Ímpar C_{2k+1}: $\chi = 3$.



C_5: Ciclo ímpar sempre vai requerer 3 cores!

3. O Caso $\chi(G) = 2$: Verificando com BFS

A verificação se um grafo pode ser colorido com apenas 2 cores equivale exatamente a verificar se ele é um grafo **Bipartido**.

Algoritmo de 2-Coloração (BFS Modificada)

1. Inicie uma Busca em Largura (BFS) a partir de um nó fonte, atribuindo-lhe a Cor 0.
2. Para cada vizinho visitado, atribua a cor oposta (Cor 1 para vizinhos de 0, Cor 0 para vizinhos de 1).
3. Ao analisar a fila, se encontrar um vizinho que já foi visitado e possui a **mesma cor** do nó atual, **o grafo NÃO é bipartido**.

3. O Caso $\chi(G) = 2$: Verificando com BFS

A verificação se um grafo pode ser colorido com apenas 2 cores equivale exatamente a verificar se ele é um grafo **Bipartido**.

Algoritmo de 2-Coloração (BFS Modificada)

1. Inicie uma Busca em Largura (BFS) a partir de um nó fonte, atribuindo-lhe a Cor 0.
2. Para cada vizinho visitado, atribua a cor oposta (Cor 1 para vizinhos de 0, Cor 0 para vizinhos de 1).
3. Ao analisar a fila, se encontrar um vizinho que já foi visitado e possui a **mesma cor** do nó atual, **o grafo NÃO é bipartido**.

Consequência de Desempenho

Colorir com 2 cores tem complexidade de tempo linear $\mathcal{O}(V + E)$. É incrivelmente rápido! Mas ao passar para 3 cores, tudo muda de figura...

O que exatamente significa a letra **N**? Ela vem de *Nondeterministic* (Não-Determinístico).

Classe P (Polinomial)

Problemas que conseguimos **resolver** do zero rapidamente (em tempo polinomial).

- **Exemplo Prático:** Encontrar o caminho mais rápido de casa para o trabalho (Google Maps). O algoritmo (ex: Dijkstra) calcula e te dá a resposta na hora.

4. A Sopa de Letrinhas: Problemas P e NP

O que exatamente significa a letra **N**? Ela vem de *Nondeterministic* (Não-Determinístico).

Classe P (Polinomial)

Problemas que conseguimos **resolver** do zero rapidamente (em tempo polinomial).

- **Exemplo Prático:** Encontrar o caminho mais rápido de casa para o trabalho (Google Maps). O algoritmo (ex: Dijkstra) calcula e te dá a resposta na hora.

Classe NP

Problemas onde achar a resposta pode ser impossível na prática, mas se alguém te der um "chute" de resposta, você consegue **verificar** se o chute está certo rapidamente.

- **Exemplo Prático:** Resolver um Sudoku gigante. Preencher tudo sozinho é difícil. Mas se um colega te entregar o tabuleiro já preenchido, você varre as linhas/colunas em segundos e atesta: "Sim, está correto!". Logo, Sudoku está em NP.

NP-Completo (O teto do NP)

São os problemas "mestres" dentro de NP. Eles conseguem simular qualquer outro problema NP.

- **Exemplo Prático:** Montar a grade horária da escola. É fácil *verificar* se uma grade sugerida não tem professores em duas salas ao mesmo tempo (logo, é NP). Mas gerar a grade do zero é tão complexo que, se você criar um algoritmo rápido pra ele, resolve todos os NP-Completos de brinde!

5. Os Chefões: NP-Completo e NP-Hard

NP-Completo (O teto do NP)

São os problemas "mestres" dentro de NP. Eles conseguem simular qualquer outro problema NP.

- **Exemplo Prático:** Montar a grade horária da escola. É fácil *verificar* se uma grade sugerida não tem professores em duas salas ao mesmo tempo (logo, é NP). Mas gerar a grade do zero é tão complexo que, se você criar um algoritmo rápido pra ele, resolve todos os NP-Completos de brinde!

NP-Hard (Os Intratáveis)

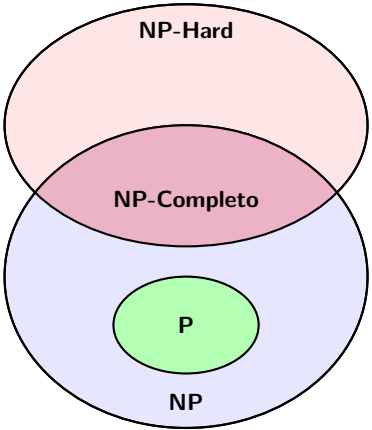
Eles são *pelo menos* tão difíceis quanto os NP-Completos, mas não exigem nem que a resposta seja verificável rapidamente!

- **Exemplo Prático:** *"Qual é a rota MAIS BARATA para o caminhão dos Correios entregar tudo?"* Se eu te der uma rota "chute", você não tem como **verificar** rapidamente se ela é a absoluta melhor sem checar infinitas outras rotas. Escapa da classe NP.

6. O Diagrama de Conjuntos

Resumindo a relação matemática das classes de complexidade:

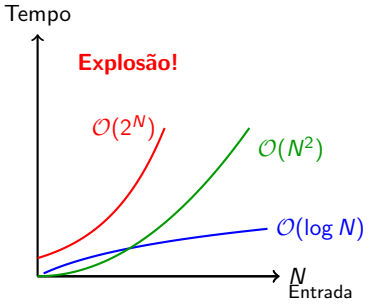
- **P:** Fáceis de resolver.
- **NP:** Fáceis de checar.
- **NP-Completo:** Os mais difíceis de resolver que são fáceis de checar.
- **NP-Hard:** Difíceis demais de resolver (e podem nem ser fáceis de checar).



7. O Abismo Computacional: Crescimento de Funções

Como o tempo de execução cresce ao aumentarmos o tamanho do problema (N)?

- **Logarítmico $\mathcal{O}(\log N)$:** Cresce muito devagar. Quase instantâneo para grandes N.
- **Polinomial $\mathcal{O}(N^2), \mathcal{O}(N^3)$:** Cresce de forma controlada. O hardware atual domina bem.
- **Exponencial $\mathcal{O}(2^N), \mathcal{O}(k^N)$:** O verdadeiro pesadelo dos problemas NP. Para um grafo médio (ex: $N = 60$), a idade do universo não seria suficiente para testar todas as possibilidades!



O Problema SAT (Satisfatibilidade Booleana)

Determinar se existe uma atribuição que torne uma fórmula booleana verdadeira. Ex: $(x_1 \vee \neg x_2) \wedge (\dots)$

O Teorema de Cook-Levin (1971) provou que o SAT foi o **primeiro problema NP-Completo**.

O Problema SAT (Satisfatibilidade Booleana)

Determinar se existe uma atribuição que torne uma fórmula booleana verdadeira. Ex: $(x_1 \vee \neg x_2) \wedge (\dots)$

O Teorema de Cook-Levin (1971) provou que o SAT foi o **primeiro problema NP-Completo**.

Todos os outros problemas NP-Completos (Caixeiro Viajante, **Coloração de Grafos**) são reduções diretas do SAT!

O Problema SAT (Satisfatibilidade Booleana)

Determinar se existe uma atribuição que torne uma fórmula booleana verdadeira. Ex: $(x_1 \vee \neg x_2) \wedge (\dots)$

O Teorema de Cook-Levin (1971) provou que o SAT foi o **primeiro problema NP-Completo**.

Todos os outros problemas NP-Completos (Caixeiro Viajante, **Coloração de Grafos**) são reduções diretas do SAT!

O que podemos ou não fazer?

- **Não podemos:** Encontrar a solução ótima exata sempre em tempo hábil. O tempo explode exponencialmente!
- **O que podemos:**
  1. **Força Bruta/Backtracking** apenas para grafos muito pequenos.
  2. **Heurísticas/Gulosos** para achar soluções "boas" de forma célere.
  3. **Aproximações** e limites matemáticos.

9. O Problema da Coloração (NP-Hard)

A transição teórica de 2 para 3 cores é onde a complexidade de algoritmos conhecidos "explode" de polinomial para exponencial.

Problema	**Complexidade**
1-colorível?	$\mathcal{O}(1)$
2-colorível?	$\mathcal{O}(V+E)$
3-colorível?	**NP-Completo!**
k-colorível ($k \geq 3$)	NP-Completo
Encontrar $\chi(G)$	NP-Hard

9. O Problema da Coloração (NP-Hard)

A transição teórica de 2 para 3 cores é onde a complexidade de algoritmos conhecidos "explode" de polinomial para exponencial.

Problema	Complexidade
1-colorível?	$\mathcal{O}(1)$
2-colorível?	$\mathcal{O}(V+E)$
3-colorível?	**NP-Completo!**
k-colorível ($k \geq 3$)	NP-Completo
Encontrar $\chi(G)$	NP-Hard

O Paradoxo da Computação

Não conhecemos nenhum algoritmo capaz de calcular o número cromático $\chi(G)$ exato para o caso geral no tempo de um polinômio.

Seja por Força Bruta ou Backtracking, o tempo de execução vai escalar vertiginosamente. Se você resolver esse desafio matematicamente provando $P = NP$, ganha 1 milhão de dólares do Instituto Clay!

10. Aplicação 1: Alocação de Registradores

Problema: Em Compiladores, os projetistas precisam mapear um número quase infinito de variáveis no código-fonte para um número restrito e muito pequeno de registradores (ex: 16 ou 32 registradores) da CPU.

10. Aplicação 1: Alocação de Registradores

Problema: Em Compiladores, os projetistas precisam mapear um número quase infinito de variáveis no código-fonte para um número restrito e muito pequeno de registradores (ex: 16 ou 32 registradores) da CPU.

Modelagem de Interferência

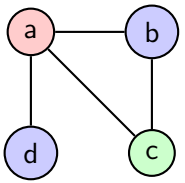
- **Vértices:** As variáveis do programa escrito.
- **Arestas:** Duas variáveis são ligadas se seus tempos de vida se sobrepõem (*liveness analysis*). Elas não podem ocupar o registrador ao mesmo tempo!
- **Cores:** Registradores físicos de silício disponíveis na placa.

10. Aplicação 1: Alocação de Registradores

Problema: Em Compiladores, os projetistas precisam mapear um número quase infinito de variáveis no código-fonte para um número restrito e muito pequeno de registradores (ex: 16 ou 32 registradores) da CPU.

Modelagem de Interferência

- **Vértices:** As variáveis do programa escrito.
- **Arestas:** Duas variáveis são ligadas se seus tempos de vida se sobrepõem (*liveness analysis*). Elas não podem ocupar o registrador ao mesmo tempo!
- **Cores:** Registradores físicos de silício disponíveis na placa.



Se $\chi(G) \leq k$, todas cabem nos registradores sem gargalos. Se não, variáveis excedentes vão para a RAM lenta, gerando *spilling*.

11. Aplicação 2: A Matemática do Sudoku

Resolver jogos de Sudoku pode ser inteiramente modelado como um exercício clássico de Coloração de Grafos.

Modelagem Abstrata

- **Vértices:** As 81 células individuais do tabuleiro 9×9.
- **Cores:** Os números disponíveis, variando de 1 a 9.
- **Arestas:** Ligam duas células vizinhas se elas se encontram na mesma linha, na mesma coluna, ou dentro do mesmo grid delimitado de 3×3.

11. Aplicação 2: A Matemática do Sudoku

Resolver jogos de Sudoku pode ser inteiramente modelado como um exercício clássico de Coloração de Grafos.

Modelagem Abstrata

- **Vértices:** As 81 células individuais do tabuleiro 9×9.
- **Cores:** Os números disponíveis, variando de 1 a 9.
- **Arestas:** Ligam duas células vizinhas se elas se encontram na mesma linha, na mesma coluna, ou dentro do mesmo grid delimitado de 3×3.

Coloração Parcial

Diferente de um grafo vazio, o Sudoku é um problema de **Coloração de Grafo Pré-preenchido**. Certos vértices (células pistas) já possuem cores cravadas.
O algoritmo precisa então completar as outras lacunas usando Backtracking para validar uma coloração 9-exata sem conflitos!

12. Aplicação 3: Timetabling (Agendamento Inteligente)

Como montar a intrincada grade horária de disciplinas da Universidade garantindo nenhum conflito para nenhum de milhares de alunos matriculados?

Modelagem do Choque de Grade

- **Vértices:** Disciplinas a ofertar ou Provas finais.
- **Arestas:** Ligam duas disciplinas **se há pelo menos um aluno em comum** matriculado em ambas as turmas.
- **Cores:** Mapeadas para cada janela (slot) de tempo disponível na semana (Ex: H1=Segunda 19h, H2=Terça 21h).

12. Aplicação 3: Timetabling (Agendamento Inteligente)

Como montar a intrincada grade horária de disciplinas da Universidade garantindo nenhum conflito para nenhum de milhares de alunos matriculados?

Modelagem do Choque de Grade

- **Vértices:** Disciplinas a ofertar ou Provas finais.
- **Arestas:** Ligam duas disciplinas **se há pelo menos um aluno em comum** matriculado em ambas as turmas.
- **Cores:** Mapeadas para cada janela (slot) de tempo disponível na semana (Ex: H1=Segunda 19h, H2=Terça 21h).

Descobrindo o Enxugamento

Rodar o algoritmo para encontrar $\chi(G)$ fornecerá a administração o **mínimo de dias ou horários** exatos necessários na semana para comportar todo o portfólio acadêmico, otimizando os dias de permanência!

13. Estimando $\chi(G)$: Limites Inferior e Superior

Calcular o $\chi(G)$ exato para centenas de nós toma uma eternidade. Os teóricos de grafos preferem focar em encontrar os limites numéricos que "cercam" a solução.

Limite Inferior: Tamanho do Clique

Se G contém dentro de si um *clique* K_ω (subgrafo completo e impenetrável de ω vértices), então categoricamente $\chi(G) \geq \omega$. Cada vértice do clique exigirá repulsão de cores máxima.

13. Estimando $\chi(G)$: Limites Inferior e Superior

Calcular o $\chi(G)$ exato para centenas de nós toma uma eternidade. Os teóricos de grafos preferem focar em encontrar os limites numéricos que "cercam" a solução.

Limite Inferior: Tamanho do Clique

Se G contém dentro de si um *clique* K_ω (subgrafo completo e impenetrável de ω vértices), então categoricamente $\chi(G) \geq \omega$. Cada vértice do clique exigirá repulsão de cores máxima.

Limite Superior: Grau Máximo

O caso do pior vértice: Um nó nunca consumirá mais cores do que seu número de conexões. Portanto, garantimos que

13. Estimando $\chi(G)$: Limites Inferior e Superior

Calcular o $\chi(G)$ exato para centenas de nós toma uma eternidade. Os teóricos de grafos preferem focar em encontrar os limites numéricos que "cercam" a solução.

Limite Inferior: Tamanho do Clique

Se G contém dentro de si um *clique* K_ω (subgrafo completo e impenetrável de ω vértices), então categoricamente $\chi(G) \geq \omega$. Cada vértice do clique exigirá repulsão de cores máxima.

Limite Superior: Grau Máximo

O caso do pior vértice: Um nó nunca consumirá mais cores do que seu número de conexões. Portanto, garantimos que

Teorema de Brooks (1941)

Para grafos conexos que escapam do pior cenário, ou seja, **não** são isoladamente cliques (K_n) nem ciclos polares ímpares, o limite superior cai amigavelmente em uma unidade para nossa sorte:

$$\chi(G) \leq \Delta(G)$$

13. Estimando $\chi(G)$: Limites Inferior e Superior

Calcular o $\chi(G)$ exato para centenas de nós toma uma eternidade. Os teóricos de grafos preferem focar em encontrar os limites numéricos que "cercam" a solução.

Limite Inferior: Tamanho do Clique

Se G contém dentro de si um *clique* K_ω (subgrafo completo e impenetrável de ω vértices), então categoricamente $\chi(G) \geq \omega$. Cada vértice do clique exigirá repulsão de cores máxima.

Limite Superior: Grau Máximo

O caso do pior vértice: Um nó nunca consumirá mais cores do que seu número de conexões. Portanto, garantimos que

Teorema de Brooks (1941)

Para grafos conexos que escapam do pior cenário, ou seja, **não** são isoladamente cliques (K_n) nem ciclos polares ímpares, o limite superior cai amigavelmente em uma unidade para nossa sorte:

$$\chi(G) \leq \Delta(G)$$

Cercamento Prático (Resumo):

$$\omega(G) \leq \chi(G) \leq \Delta(G) + 1$$

14. Welsh-Powell: A Heurística Gulosa Clássica

Uma heurística não busca a perfeição matemática, mas sim uma aproximação extremamente célere. A abordagem gulosa tenta colorir de forma voraz, pintando antes os nós "mais problemáticos" da rede (maior número de laços).

O Algoritmo Clássico

1. Varrer o grafo calculando o grau total adjacente de todos os vértices.
2. Realizar o *sort* ordenando os vértices em ordem estritamente **decrescente** de grau.
3. Iterando sobre cada vértice na nova ordem disposta:
   - Atribua a ele a **primeira cor da paleta** (menor inteiro: 0, 1, 2...) que não choque com nenhum dos seus vizinhos já visitados.

14. Welsh-Powell: A Heurística Gulosa Clássica

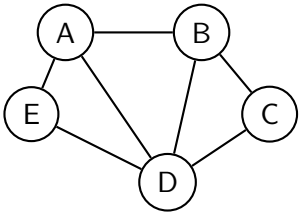
Uma heurística não busca a perfeição matemática, mas sim uma aproximação extremamente célere. A abordagem gulosa tenta colorir de forma voraz, pintando antes os nós "mais problemáticos" da rede (maior número de laços).

O Algoritmo Clássico

1. Varrer o grafo calculando o grau total adjacente de todos os vértices.
2. Realizar o *sort* ordenando os vértices em ordem estritamente **decrescente** de grau.
3. Iterando sobre cada vértice na nova ordem disposta:
   - Atribua a ele a **primeira cor da paleta** (menor inteiro: 0, 1, 2...) que não choque com nenhum dos seus vizinhos já visitados.

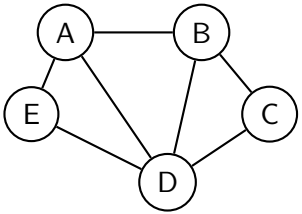
Alerta de Projetista: Funciona super bem alcançando tetos próximos do ótimo na fração de milissegundos, mas **não há qualquer garantia do $\chi(G)$ ótimo**! Seu score fica à mercê das decisões de ordenação.

15. Rastreando Welsh-Powell na Prática



Passo 1 (Graus): D=4, A=3, B=3, C=2, E=2.
Passo 2 (Fila): D, A, B, C, E. (Empates ord. alfabética)

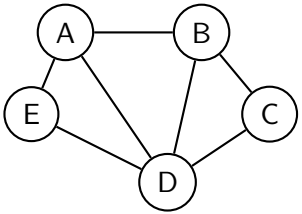
15. Rastreando Welsh-Powell na Prática



Passo 1 (Graus): D=4, A=3, B=3, C=2, E=2.
Passo 2 (Fila): D, A, B, C, E. (Empates ord. alfabética)

Nó	Vizinhos Coloridos	Restrições	Cor da Paleta

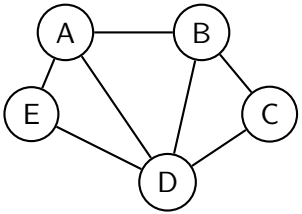
15. Rastreando Welsh-Powell na Prática



Passo 1 (Graus): D=4, A=3, B=3, C=2, E=2.
Passo 2 (Fila): D, A, B, C, E. (Empates ord. alfabética)

Nó	Vizinhos Coloridos	Restrições	Cor da Paleta
D	—	—	**0** (Vermelho)

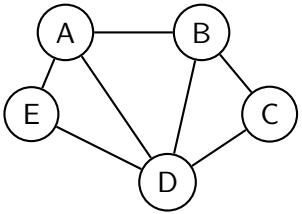
15. Rastreando Welsh-Powell na Prática



Passo 1 (Graus): D=4, A=3, B=3, C=2, E=2.
Passo 2 (Fila): D, A, B, C, E. (Empates ord. alfabética)

Nó	Vizinhos Coloridos	Restrições	Cor da Paleta
D	—	—	**0** (Vermelho)
A	D (Cor 0)	$\{0\}$	**1** (Azul)

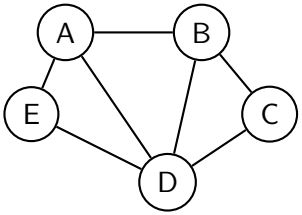
15. Rastreando Welsh-Powell na Prática



Passo 1 (Graus): D=4, A=3, B=3, C=2, E=2.
Passo 2 (Fila): D, A, B, C, E. (Empates ord. alfabética)

Nó	Vizinhos Coloridos	Restrições	Cor da Paleta
D	—	—	**0** (Vermelho)
A	D (Cor 0)	{0}	**1** (Azul)
B	D (Cor 0), A (Cor 1)	{0, 1}	**2** (Verde)

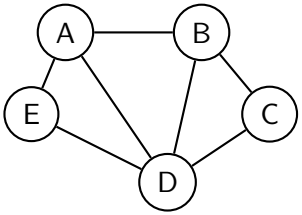
15. Rastreando Welsh-Powell na Prática



Passo 1 (Graus): D=4, A=3, B=3, C=2, E=2.
Passo 2 (Fila): D, A, B, C, E. (Empates ord. alfabética)

Nó	Vizinhos Coloridos	Restrições	Cor da Paleta
D	—	—	**0** (Vermelho)
A	D (Cor 0)	{0}	**1** (Azul)
B	D (Cor 0), A (Cor 1)	{0, 1}	**2** (Verde)
C	D (Cor 0), B (Cor 2)	{0, 2}	**1** (Azul livre!)

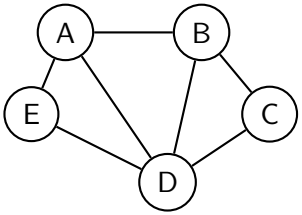
15. Rastreando Welsh-Powell na Prática



Passo 1 (Graus): D=4, A=3, B=3, C=2, E=2.
Passo 2 (Fila): D, A, B, C, E. (Empates ord. alfabética)

Nó	Vizinhos Coloridos	Restrições	Cor da Paleta
D	—	—	**0** (Vermelho)
A	D (Cor 0)	{0}	**1** (Azul)
B	D (Cor 0), A (Cor 1)	{0, 1}	**2** (Verde)
C	D (Cor 0), B (Cor 2)	{0, 2}	**1** (Azul livre!)
E	D (Cor 0), A (Cor 1)	{0, 1}	**2** (Verde livre!)

15. Rastreando Welsh-Powell na Prática



Passo 1 (Graus): D=4, A=3, B=3, C=2, E=2.
Passo 2 (Fila): D, A, B, C, E. (Empates ord. alfabética)

Nó	Vizinhos Coloridos	Restrições	Cor da Paleta
D	—	—	**0** (Vermelho)
A	D (Cor 0)	{0}	**1** (Azul)
B	D (Cor 0), A (Cor 1)	{0, 1}	**2** (Verde)
C	D (Cor 0), B (Cor 2)	{0, 2}	**1** (Azul livre!)
E	D (Cor 0), A (Cor 1)	{0, 1}	**2** (Verde livre!)

Resultado do Guloso: 3 cores. Neste caso fortuito, ele tropeçou no número ótimo matemático.

16. Implementação no Java – Welsh-Powell

```java
public static int[] greedyColoring(int V, List<List<Integer>> adj) {
    int[] color = new int[V];
    Arrays.fill(color, -1);

    // Sort array of object encapsulando 'indices pelo tamanho da adjac ncia (Grau)
    Integer[] order = new Integer[V];
    for (int i = 0; i < V; i++) order[i] = i;
    Arrays.sort(order, (a, b) -> adj.get(b).size() - adj.get(a).size());

    boolean[] available = new boolean[V];

    for (int u : order) {
        Arrays.fill(available, true); // Zera tabela verdade da paleta
        for (int viz : adj.get(u))
            if (color[viz] != -1)
                available[color[viz]] = false; // Flag inibe cor se o vizinho j'a
                    usar

        for (int c = 0; c < V; c++) // Varre array de boolean para caçar 1   gap
            livre
            if (available[c]) { color[u] = c; break; }
    }
```

17. A Heurística Avançada: DSatur (Degree of Saturation)

O galcanhar de Aquiles do Welsh-Powell é sua natureza **estática** (a ordenação feita no início nunca mais se adequa à realidade da malha que está sendo pintada). O algoritmo **DSatur** (Brelaz, 1979) resolve inúmeras falhas do guloso clássico operando de maneira **dinâmica**.

O que compõe o Grau de Saturação?

O grau de saturação de um nó mede o seu nível de estresse restritivo: É a métrica contando o número de **cores distintas** que a vizinhança do vértice já consolidou em sua volta.

17. A Heurística Avançada: DSatur (Degree of Saturation)

O galcanhar de Aquiles do Welsh-Powell é sua natureza **estática** (a ordenação feita no início nunca mais se adequa à realidade da malha que está sendo pintada). O algoritmo **DSatur** (Brelaz, 1979) resolve inúmeras falhas do guloso clássico operando de maneira **dinâmica**.

O que compõe o Grau de Saturação?

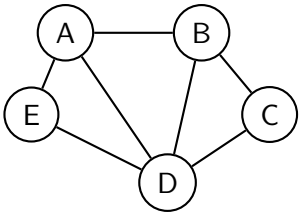
O grau de saturação de um nó mede o seu nível de estresse restritivo: É a métrica contando o número de **cores distintas** que a vizinhança do vértice já consolidou em sua volta.

O Protocolo Dinâmico DSatur

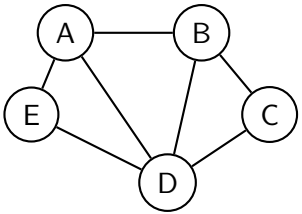
1. Para o início do laço, o vértice de maior saturação será o nó de **maior grau do grafo** (desempate global).
2. No laço principal: Escolha dinamicamente o vértice não colorido exibindo o **maior grau de saturação momentâneo**. (Desempate pelo maior grau residual se necessário).
3. Atribua-lhe imediatamente a menor cor paletizável disponível para não esbarrar nos vizinhos.

Passo-a-Passo: O DSatur em Ação

Inicialização: Maior grau é D(4). Pinta D de **Cor 0**.



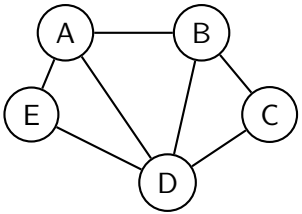
Passo-a-Passo: O DSatur em Ação



Inicialização: Maior grau é D(4). Pinta D de **Cor 0**.

Saturação 1: A(1), B(1), C(1), E(1). Empate de saturação! Desempate por grau: A(3) e B(3). Escolhemos A. Vizinho: D(0). Pinta A de **Cor 1**.

Passo-a-Passo: O DSatur em Ação

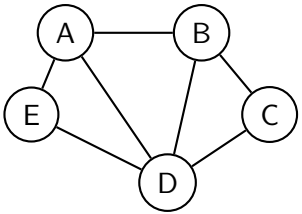


Inicialização: Maior grau é D(4). Pinta D de **Cor 0**.

Saturação 1: A(1), B(1), C(1), E(1). Empate de saturação! Desempate por grau: A(3) e B(3). Escolhemos A. Vizinho: D(0). Pinta A de **Cor 1**.

Saturação 2: B e E veem D(0) e A(1) $\rightarrow$ **Sat=2**. C vê D(0) $\rightarrow$ Sat=1. Empate (B e E). Grau B(3) ¿ Grau E(2). Escolhe B. Pinta B de **Cor 2**.

Passo-a-Passo: O DSatur em Ação



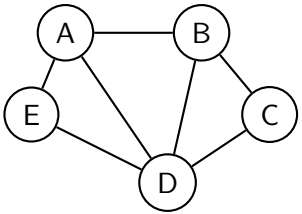
Inicialização: Maior grau é D(4). Pinta D de **Cor 0**.

Saturação 1: A(1), B(1), C(1), E(1). Empate de saturação! Desempate por grau: A(3) e B(3). Escolhemos A. Vizinho: D(0). Pinta A de **Cor 1**.

Saturação 2: B e E veem D(0) e A(1) $\rightarrow$ **Sat=2**. C vê D(0) $\rightarrow$ Sat=1. Empate (B e E). Grau B(3) ¿ Grau E(2). Escolhe B. Pinta B de **Cor 2**.

Saturação 3: C vê D(0), B(2) $\rightarrow$ **Sat=2**. E vê D(0), A(1) $\rightarrow$ **Sat=2**. Empate! Escolhe C. Pinta de **Cor 1** (livre p/ ele).

Passo-a-Passo: O DSatur em Ação



Inicialização: Maior grau é D(4). Pinta D de **Cor 0**.

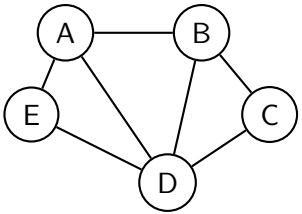
Saturação 1: A(1), B(1), C(1), E(1). Empate de saturação! Desempate por grau: A(3) e B(3). Escolhemos A. Vizinho: D(0). Pinta A de **Cor 1**.

Saturação 2: B e E veem D(0) e A(1) $\rightarrow$ **Sat=2**. C vê D(0) $\rightarrow$ Sat=1. Empate (B e E). Grau B(3) ¿ Grau E(2). Escolhe B. Pinta B de **Cor 2**.

Saturação 3: C vê D(0), B(2) $\rightarrow$ **Sat=2**. E vê D(0), A(1) $\rightarrow$ **Sat=2**. Empate! Escolhe C. Pinta de **Cor 1** (livre p/ ele).

Final: Restou E com **Sat=2**. Cores na vizinhança: {0,1}. Pinta E de **Cor 2**.

Passo-a-Passo: O DSatur em Ação



Inicialização: Maior grau é D(4). Pinta D de **Cor 0**.

Saturação 1: A(1), B(1), C(1), E(1). Empate de saturação! Desempate por grau: A(3) e B(3). Escolhemos A. Vizinho: D(0). Pinta A de **Cor 1**.

Saturação 2: B e E veem D(0) e A(1) $\rightarrow$ **Sat=2**. C vê D(0) $\rightarrow$ Sat=1. Empate (B e E). Grau B(3) ¿ Grau E(2). Escolhe B. Pinta B de **Cor 2**.

Saturação 3: C vê D(0), B(2) $\rightarrow$ **Sat=2**. E vê D(0), A(1) $\rightarrow$ **Sat=2**. Empate! Escolhe C. Pinta de **Cor 1** (livre p/ ele).

Final: Restou E com **Sat=2**. Cores na vizinhança: {0,1}. Pinta E de **Cor 2**.

Observe: a ordem de escolha dos vértices mudou dinamicamente a cada passo, priorizando quem estava mais restrito por cores!

18. Solução Exata Pura (DFS com Backtracking)

Para topologias com V pequeno e enxuto ($\leq 20 \sim 25$), ou quando é de fundamental criticidade acadêmica possuir o valor absoluto garantido, apela-se ao uso das profundezas da Busca Backtracking. Ela investigará impiedosamente a árvore de decisão tentando exaurir os conflitos, retrocedendo as decisões em que encontra paredes.

```
static boolean backtrack(List<List<Integer>> adj, int[] color, int u, int k) {
    if (u == color.length) return true; // Sucesso!  ltimo  n'o repousou
        pacificamente.

    for (int c = 0; c < k; c++) { // Empurra cada cor da paleta limitada 'k' ao node
         'u'
        if (isSafe(adj, color, u, c)) { // Consulta integridade estrutural
            color[u] = c; // Efetiva mutação provis'oria no tecido
            if (backtrack(adj, color, u + 1, k)) return true; // Avante!
            color[u] = -1; // Dead-End (Beco Sem Sa'ida). Limpe e inicie Backtracking
                .
        }
    }
    return false; // Este braço do universo estilhaçou as esperanças. Suba a call-
        stack.
}
```

19. A Matemática Oculta: Polinômio Cromático $P(G, k)$

Além da busca obstinada pelo menor índice cromático, os pesquisadores expandem a visão interrogando-se: **De quantas maneiras criativas distintas** o grafo G poderia ser pintado se alguém me ofertasse um balde generoso exato contendo k tintas em mãos? A resposta é ditada rigorosamente por um polinômio $P(G, k)$, condicionado estruturalmente a k.

Mapeamentos Notáveis

- **Grafo Vazio:** Equivale a $P(G, k) = k^V$ (Cores independentes e alienadas do seu redor)
- **Grafo Completo K_n:** $P(K_n, k) = k \cdot (k-1) \dots (k-V+1)$ (Progressão decrescente dura)
- **Árvores Esparsas:** $P(Tree, k) = k \cdot (k-1)^{V-1}$ (Apenas um elo parental para evitar ao longo da

19. A Matemática Oculta: Polinômio Cromático $P(G, k)$

Além da busca obstinada pelo menor índice cromático, os pesquisadores expandem a visão interrogando-se: **De quantas maneiras criativas distintas** o grafo G poderia ser pintado se alguém me ofertasse um balde generoso exato contendo k tintas em mãos? A resposta é ditada rigorosamente por um polinômio $P(G, k)$, condicionado estruturalmente a k.

Mapeamentos Notáveis

- **Grafo Vazio:** Equivale a $P(G, k) = k^V$ (Cores independentes e alienadas do seu redor)
- **Grafo Completo K_n:** $P(K_n, k) = k \cdot (k-1) \dots (k-V+1)$ (Progressão decrescente dura)
- **Árvores Esparsas:** $P(Tree, k) = k \cdot (k-1)^{V-1}$ (Apenas um elo parental para evitar ao longo da

Onde reside o Santo Graal?

Toda essa álgebra culmina numa regra fantástica. Injetamos variados 'k' cores ali... Se o polinômio resultar positivo $P(G, k) > 0$, concluímos ser possível modelar o grafo! E o $\chi(G)$? Ora, ele é desmascarado como sendo o **menor numeral inteiro** em k a não zerar a equação!

20. O Assustador Teorema das Quatro Cores

O Postulado (Appel & Haken, 1976)

Todo grafo planar pode ser colorido com no máximo 4 cores.

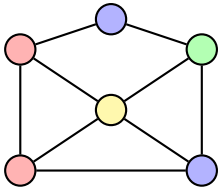
Trocando em miúdos: $\chi(G) \leq 4$ para qualquer grafo desenhável sem cruzamento de arestas.

O Postulado (Appel & Haken, 1976)

Todo grafo planar pode ser colorido com no máximo 4 cores.
Trocando em miúdos: $\chi(G) \leq 4$ para qualquer grafo desenhável sem cruzamento de arestas.

O que são Grafos Planares?

Podem ser desenhados no plano 2D sem que as arestas se cruzem. O modelo clássico de planaridade são as fronteiras de um mapa geográfico.



Limite máximo de apenas 4 potes de tinta para pintar todo o mapa-múndi.

20. O Assustador Teorema das Quatro Cores

O Postulado (Appel & Haken, 1976)

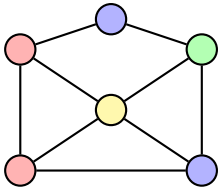
Todo grafo planar pode ser colorido com no máximo 4 cores.
Trocando em miúdos: $\chi(G) \leq 4$ para qualquer grafo desenhável sem cruzamento de arestas.

O que são Grafos Planares?

Podem ser desenhados no plano 2D sem que as arestas se cruzem. O modelo clássico de planaridade são as fronteiras de um mapa geográfico.

A Prova Computacional

A demonstração rasgou paradigmas: usou supercomputadores para checar exaustivamente 1.936 configurações de conflitos (impossível para humanos). Foi a primeira prova matemática assistida por computador.



Limite máximo de apenas 4 potes de tinta para pintar todo o mapa-múndi.

21. Coloração de Arestas e o Teorema de Vizing

Ao invés de vértices, podemos colorir as **arestas**. A regra é: arestas que compartilham o mesmo vértice não podem ter a mesma cor. Isso define o Índice Cromático da Aresta $\chi'(G)$.

Ao invés de vértices, podemos colorir as **arestas**. A regra é: arestas que compartilham o mesmo vértice não podem ter a mesma cor. Isso define o Índice Cromático da Aresta $\chi'(G)$.

Teorema de Vizing (1964)

Para grafos simples (sem laços ou arestas múltiplas), o índice cromático da aresta é incrivelmente restrito:

$$\chi'(G) \in \{\Delta(G), \Delta(G) + 1\}$$

21. Coloração de Arestas e o Teorema de Vizing

Ao invés de vértices, podemos colorir as **arestas**. A regra é: arestas que compartilham o mesmo vértice não podem ter a mesma cor. Isso define o Índice Cromático da Aresta $\chi'(G)$.

Teorema de Vizing (1964)

Para grafos simples (sem laços ou arestas múltiplas), o índice cromático da aresta é incrivelmente restrito:

$$\chi'(G) \in \{\Delta(G), \Delta(G) + 1\}$$

Aplicação: Torneio Round-Robin

- **Vértices:** Equipes.
- **Arestas:** Jogos entre as equipes.
- **Cores:** Rodadas (Dias de jogo).

Quantos dias são necessários para que todos joguem contra todos (K_n) sem que um time jogue duas vezes no mesmo dia?

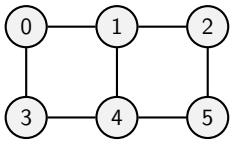
Pelo Teorema de Vizing:
$\rightarrow n - 1$ dias (se n for par)
$\rightarrow n$ dias (se n for ímpar)

Exercite a mente visual classificada para extrair o Número Cromático $\chi(G)$ exato que impera em cada formação abaixo citada:

1. Uma malha grade (grid retangular) puramente bidimensional em proporções de $N \times N$.
2. Uma espessa Árvore de Nível de 1.000.000 de vértices com raízes bifurcadas infinitamente em leque.
3. Uma formação de centro concentrada em estrela tipo S_{10} (um vértice nevrálgico no cerne suportando exatas 10 folhas penduradas isoladamente).
4. Um anel vicioso circular de laços montados num elo longo com 7 vértices escorrendo de ponta a ponta (Polígono Fechado C_7).

Laboratório 2: Batalha de Algoritmos (Welsh-Powell Manual)

Coloque o jaleco e simule mecanicamente a destreza algorítmica da gulosa no corpo de malha à direita:



1. Mapeie minuciosamente o volume de conexão de cada elemento (Os graus inerentes).
2. Produza a lista serial ordenando os envolvidos via critério reverso (Grau do superior decaindo linearmente). Em caso colisão equitativa, privilegie o rótulo numérico nativo menor (Desempate 1 ganha do 2).
3. Caminhe pela fita recém orquestrada enfiando a cor de mais baixo valor da prateleira.
4. Responda: O método poupou quantas cores? Essa quantia atinge o mérito de Ouro (Teto matemático da melhor opção $\chi(G)$ real desse desenho) ou o Welsh engasgou

Laboratório 3: Desafio Profissional - O Corretor da Matrícula

O Desastre Administrativo da Universidade

Você como recém-graduado arquiteto do sistema deve orquestrar as provas semestrais envolvendo 5 grandes troncos do currículo (A, B, C, D, E). A catástrofe de software passada sobrepôs indevidamente turnos cruzados. Os logaritmos rastrearam alunos duplicados gerando colisões inaceitáveis da base acadêmica: A colide com B e C. B esbarra em D. A turma de C também divide o público de D. Finalmente as cadeiras exclusivas de D retém interessados do tronco de E. (Arestas de conflito: A-B, A-C, B-D, C-D, D-E).

1. Use folha e papel para materializar a visão relacional num Grafo Conectivo.
2. Calcule intelectualmente e descubra o mínimo enxuto de horários (turnos / slots solares) obrigatórios a se provisionar na tabela para esgotar o estresse operacional dos choques de provas.
3. Prove seu raciocínio ofertando um quadro de designação real alocando-os.

Laboratório 3: Desafio Profissional - O Corretor da Matrícula

O Desastre Administrativo da Universidade

Você como recém-graduado arquiteto do sistema deve orquestrar as provas semestrais envolvendo 5 grandes troncos do currículo (A, B, C, D, E). A catástrofe de software passada sobrepôs indevidamente turnos cruzados. Os logaritmos rastrearam alunos duplicados gerando colisões inaceitáveis da base acadêmica: A colide com B e C. B esbarra em D. A turma de C também divide o público de D. Finalmente as cadeiras exclusivas de D retém interessados do tronco de E. (Arestas de conflito: A-B, A-C, B-D, C-D, D-E).

1. Use folha e papel para materializar a visão relacional num Grafo Conectivo.
2. Calcule intelectualmente e descubra o mínimo enxuto de horários (turnos / slots solares) obrigatórios a se provisionar na tabela para esgotar o estresse operacional dos choques de provas.
3. Prove seu raciocínio ofertando um quadro de designação real alocando-os.

Resposta Reveladora de Projeto Estrutural

Após desenhado, presenciamos o bloco fundamental sendo selado pela forma de um ciclo

Mapeamento Recapitulativo da Arte da Coloração

Algoritmo	Complexidade	Características
BFS Modificada (2-Bipartição)	$\mathcal{O}(V+E)$	Exato garantido, mas apenas para $k=2$
Welsh-Powell (Heurística Gulosa)	$\mathcal{O}(V \log V)$	Usa $\leq \Delta+1$ cores. Não garante o ótimo.
DSatur (Grau de Saturação)	$\mathcal{O}(V^2)$	Dinâmico e poderoso. Excelente na prática.
Backtracking (Força Bruta)	$\mathcal{O}(k^V)$	Acerto perfeito. Inviável para V grandes.

Teoremas e Aplicações Essenciais

- **Aplicações:** Alocação de registradores (CPU), Sudoku, e Agendamento (Timetabling).
- **Teorema de Brooks:** $\chi(G) \leq \Delta(G)$ (exceto grafos completos e ciclos ímpares).
- **Teorema das Quatro Cores:** Todo grafo planar pode ser colorido com ≤ 4 cores.
- **Teorema de Vizing:** O índice cromático das *arestas* é $\Delta(G)$ ou $\Delta(G)+1$.